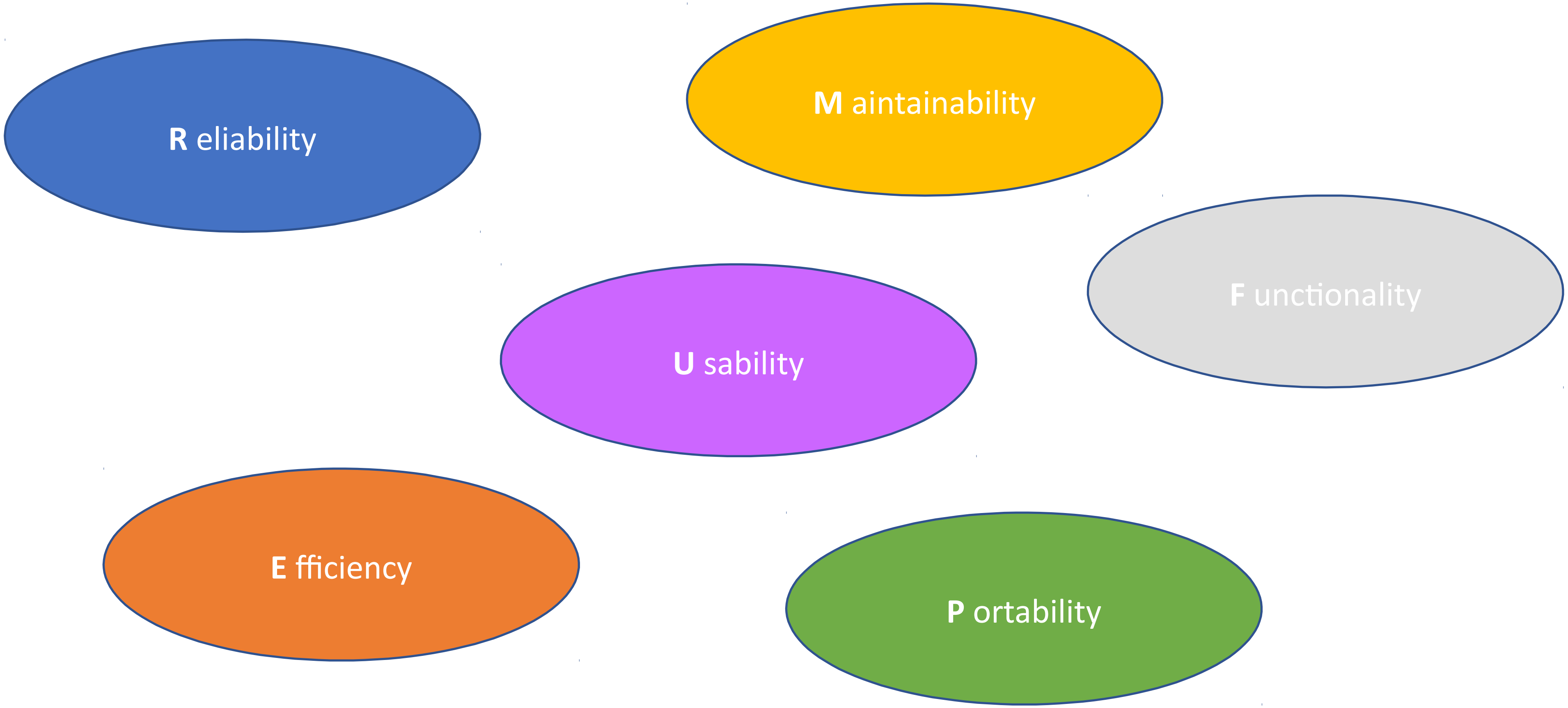
12-FACTOR-APP,
BUILD CLOUD READY APPLICATIONS

2018-02-01, UWE EISELE / DIETER BAIER, UAS FRANKFURT

NOVATEC | FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Schönbuch

The Twelve-Factor App (https://12factor.net)
SW – Quality Model ISO/IEC 9126



Reliability

Maintainability

Functionality

Usability

Efficiency

Portability

The Twelve-Factor App (https://12factor.net)

(From the manifesto itself)

Methodology for building SAAS apps that

- Use declarative formats for setup automation
- Have a clean contract with the underlying operating system
- Are suitable for deployment on modern cloud platforms
- Minimize divergence between development and production
- Can scale up without significant changes to tooling, architecture or development practices

Can be applied to apps written in any programming language, and which use any combination of backing services.

It is a triangulation on ideal practices for app development, paying particular attention to the dynamics of the organic growth of an app over time, the dynamics of collaboration between developers working on the app's codebase, and avoiding the cost of software erosion

The Twelve-Factor App (https://12factor.net)

I
Codebase

One codebase, many deploys

II
Dependencies

Never relies on implicit existence of system-wide packages

III
Config

Strict seperation of code and config

IV
Backing Services

Doesn‘t distinct between a local or remote service

V
Build, release, run

Strict seperation of the three dev-steps – no way to change the app during runtime

VI
Processes

Stateless and share nothing

VII
Port binding

Is completely self-contained and doesn‘t rely on runtime services

VIII
Concurrency

It‘s processes are ‚first class citizens‘

IX
Disposability

Can be started or stopped at a moment‘s notice

X
Dev/prod parity

Use the same backing services in all stages

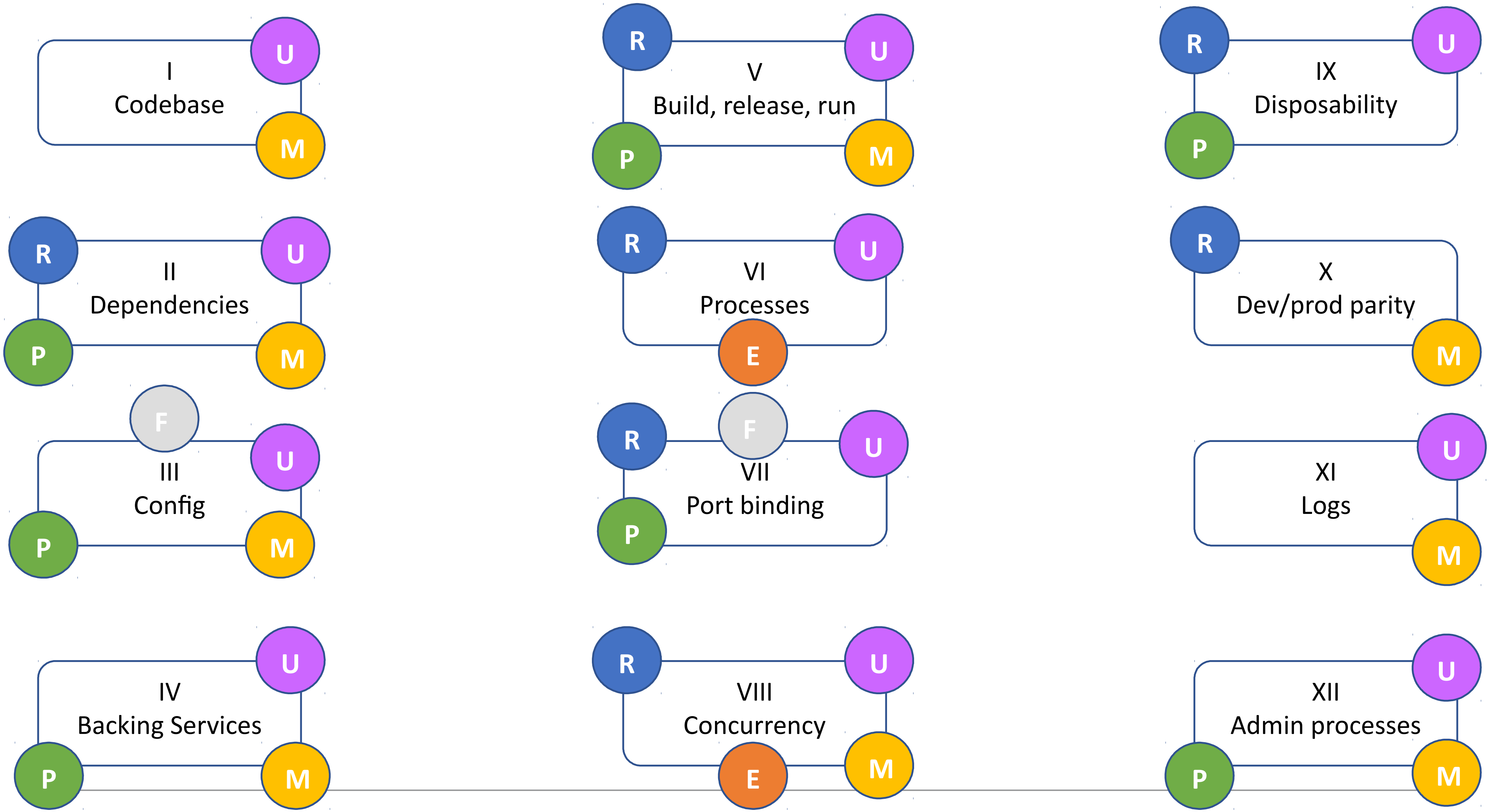
XI
Logs

Logs to the output stream and doesn‘t concern of ist storage

XII
Admin processes

Provide possibilities to run run one-time-tasks within the same environment

The Twelve-Factor App (https://12factor.net)



I
Codebase
U M

R U
II
Dependencies
P M

F
III
Config
U P M

IV
Backing Services
U P M

R U
V
Build, release, run
P M

R U
VI
Processes
E

R F U
VII
Port binding
P

R U
VIII
Concurrency
E M

R U
IX
Disposability
P

R
X
Dev/prod parity
M

XI
Logs
U M

XII
Admin processes
U P M

The Twelve-Factor App – An Example

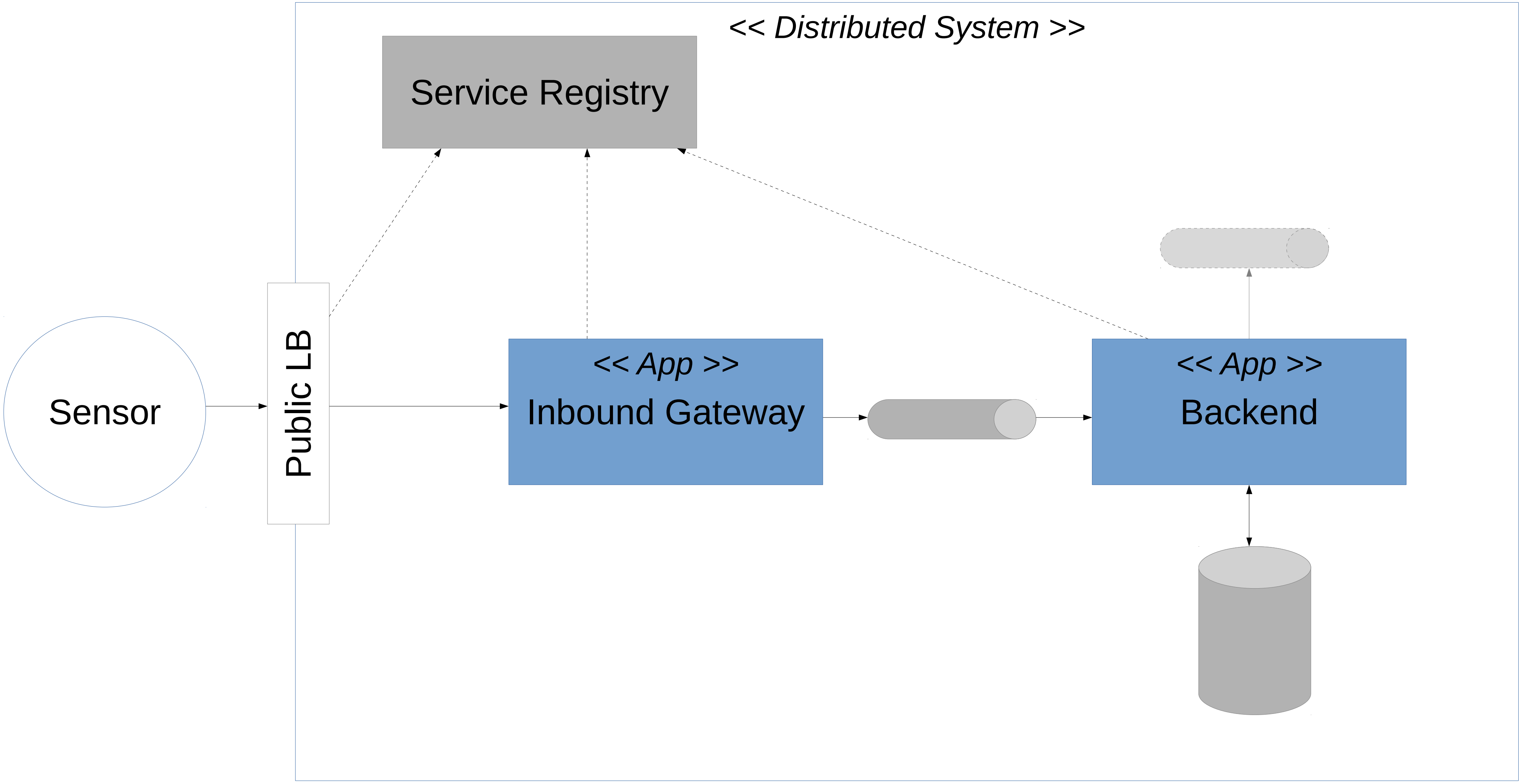
https://github.com/ntuas

The smart fridge – an example from IoT

- The fridge recognizes new products put into it
- The fridge recognizes if products where taken from it
- The fridge orders missing products on demand

The Twelve-Factor App – An Example

https://github.com/ntuas



<< Distributed System >>

Service Registry

Sensor

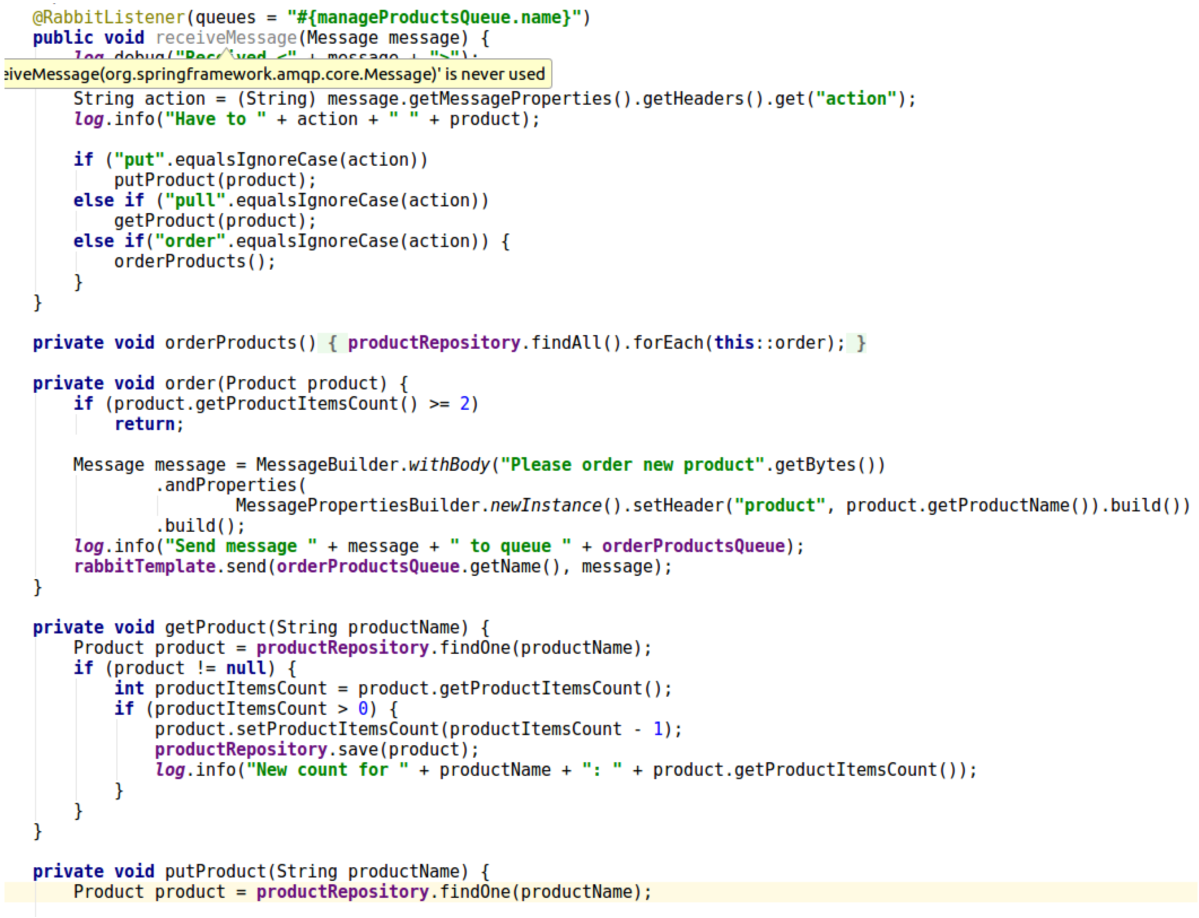
Public LB

<< App >>
Inbound Gateway

<< App >>
Backend

The Twelve-Factor App – Code Examples

https://github.com/ntuas



```
@RabbitListener(queues = "#{manageProductsQueue.name}")
public void receiveMessage(Message message) {
    log.debug("Received <" + message + ">");
eiveMessage(org.springframework.amqp.core.Message)' is never used
    String action = (String) message.getMessageProperties().getHeaders().get("action");
    log.info("Have to " + action + " " + product);

    if ("put".equalsIgnoreCase(action))
        putProduct(product);
    else if ("pull".equalsIgnoreCase(action))
        getProduct(product);
    else if("order".equalsIgnoreCase(action)) {
        orderProducts();
    }
}

private void orderProducts() { productRepository.findAll().forEach(this::order); }

private void order(Product product) {
    if (product.getProductItemsCount() >= 2)
        return;

    Message message = MessageBuilder.withBody("Please order new product".getBytes())
            .andProperties(
                    MessagePropertiesBuilder.newInstance().setHeader("product", product.getProductName()).build())
            .build();
    log.info("Send message " + message + " to queue " + orderProductsQueue);
    rabbitTemplate.send(orderProductsQueue.getName(), message);
}

private void getProduct(String productName) {
    Product product = productRepository.findOne(productName);
    if (product != null) {
        int productItemsCount = product.getProductItemsCount();
        if (productItemsCount > 0) {
            product.setProductItemsCount(productItemsCount - 1);
            productRepository.save(product);
            log.info("New count for " + productName + ": " + product.getProductItemsCount());
        }
    }
}

private void putProduct(String productName) {
    Product product = productRepository.findOne(productName);
```

The Twelve-Factor App - Outlook

Still missing something?

III
Config

V
Build, release, run

XI
Logs

XII
Admin processes

12-FACTOR-APP, BUILD CLOUD READY APPLICATIONS

2018-02-01, UAS FRANKFURT

DIETER BAIER (DIETER.BAIER@NOVATEC-GMBH.DE)
UWE EISELE (UWE.EISELE@NOVATEC-GMBH.DE)

NOVATEC | FRANKFURT UNIVERSITY OF APPLIED SCIENCES